A Goal-oriented Approach for Designing Collaboration Processes

Bahar Ghadiri Bashardoost
University of Toronto
ghadiri@cs.toronto.edu

Kelly Lyons
University of Toronto
klyons@cs.toronto.edu

Rock Leung
SAP
Rock.leung@sap.com

Abstract

Although Group Support Systems (GSSs) have been studied and developed for many years, they are not widely used in practice. One of the challenges of using GSSs is that extensive knowledge and expertise are needed in order to design effective collaboration processes. Research has sought to make this knowledge available to practitioners by introducing modules of collaboration called thinkLets; however, facilitation expertise is often still needed to select the correct thinkLet(s) for a given group activity. We propose an approach for designing collaboration processes that makes use of multi-criteria decision making, decision-tree models, and goal programming. The effectiveness of our approach was evaluated by comparing collaboration process designs developed by people using our prototype with those by people using detailed written documentation. We found that our approach significantly improves the efficiency of the collaboration process design procedure by guiding practitioners in choosing the best combination of thinkLets.

1. Introduction

Increasingly, collaboration is required to solve the complex problems of today. The wide range of information technologies from e-mail to Group Support Systems (GSS) are key enablers for effective and efficient collaboration [3]. Group support systems are among the few e-collaboration technologies that provide a structure that can support group development and productive outcomes [22]. They have also been shown to increase a team's productivity in many circumstances [15,25,10]. Studies suggest that GSS tools, when used properly, improve efficiency, effectiveness, usability, consensus building and satisfaction in comparison with manual methods [14].

Nunamaker et al., propose that one of the basic principles of successful use of GSS tools is to have a well defined goal and to be able to design a GSS session which can clearly advance the team members toward accomplishing that goal [23]. However, designing collaboration processes that effectively harness the knowledge of multiple people in collaborative efforts and push them towards achieving a shared goal is not an easy task [20].

Collaboration engineering is a field of study established to encapsulate key facilitation techniques to support the execution of collaborative activities by teams that do not have professional facilitation support [18]. One of the key concepts introduced through collaboration engineering is the notion of repeatable processes called *thinkLets*. Each thinkLet includes: 1) a description of a tool or GSS component that can be used; 2) details regarding how to configure the tool; and 3) a script which contains instructions that should be given to the decision-making group in using the tool during the session [7]. It has been shown that a small change in any of these three pieces of information can change the outcome of the collaboration process significantly [21,7]. Researchers argue that by making these three pieces of information available through thinkLets to non-experienced facilitators, decision-making groups can produce predictable outcomes [7].

Detailed documentation is made available for each thinkLet that summarizes the criteria for deciding when and when not to use that thinkLet. Among these criteria, the documentation highlights the input required by the thinkLet (e.g., large number of brainstorming comments) and the structure of thinkLet's output (e.g., a set of comments organized by discussion topic) [7].

ThinkLets can be sequenced together to produce a collaboration process such that the output of one thinkLet is used as an input for the next thinkLet. A combination of two thinkLets is called *Tricky* when the output of the first thinkLet is not fully compatible with the input of the second thinkLet [19]. If the output of the first thinkLet is not at all compatible with the input of the second thinkLet the combination is called *Impossible* [19]. Otherwise the combination of the two thinkLets is called *Excellent* [19].

Toolkits have been developed to guide practitioners (domain experts with scant knowledge of collaboration engineering) through the selection of thinkLets [20,29]. Nevertheless, even toolkits that have been developed based on collaboration engineering principles still

require a facilitation expert to decide among alternatives at various stages of the process design.

In this paper we introduce a design approach and accompanying tool called CP-Dez which can guide practitioners in the design of a collaboration process by helping them to choose the right sequence of thinkLets for a given decision task without requiring expert intervention. This is possible because we use a decision tree and multiple criteria decision making (MCDM) techniques including goal programming. A decision tree is used to classify the different thinkLets. In some cases, the best thinkLet can be identified by traversing the decision tree. However, when the classification rules of the decision tree are not sufficient to identify a single thinkLet, goal-programming combined with additive value function rules are used to help a practitioner make trade-offs and choose a single thinkLet that most satisfies the group goals. Our contributions can be summarized as follows:

- We propose a model to capture and represent the knowledge of collaboration engineering. Our model is easy to extend and maintain by individuals or collaboratively by a community of collaboration engineers.
- We propose a method that, given a set of domain-specific criteria, extracts a high-quality collaboration process from the model. This method which is based on MCDM can be used by practitioners to design a collaboration process.
- We created a prototype based on our proposed design and ran a preliminary user study to evaluate the effectiveness of our approach. The evaluation was performed by comparing participants who used our prototype to identify thinkLets for a collaborative activity, with those who used the thinkLet book [5]. The results suggest that our approach can significantly improve the efficiency of the collaboration design procedure.

In section 2, we delve deeper into the motivation for our design and overview related work. In section 3, we present the details of our design. In section 4, we describe the study used to evaluate the effectiveness of our prototype. In section 5, the results of the study are presented. Finally, in section 6, we identify limitations of our approach and suggest areas for future research.

2. Background and related work

Designing collaboration processes requires knowledge that is not easily accessible to practitioners who are domain experts but not experienced facilitators. Consequently, in order to exploit the benefits of GSS tools, teams rely on skilled facilitators to design a collaboration process that minimizes distraction, focuses attention towards achieving a shared goal [8] and helps group members align their individual goals with those of the group [9].

While a group can significantly benefit from a facilitator-led GSS session, professional facilitators are expensive and difficult to retain. In addition, their extensive set of skills is difficult to transfer and thus an organization can become reliant on them [6]. In order to guide practitioners and inexperienced facilitators to design collaboration processes, toolkits have been developed to help find the best thinkLet for the collaboration task at hand [20,29].

The toolkit by Kolfschoten, et al. [20] tries to narrow down the choice of thinkLets by filtering them based on their result, input, special characteristics and, most importantly, the desired pattern of collaboration. In this toolkit, the practitioner chooses from among the filtered list of thinkLets, which still requires a considerable amount of knowledge about collaboration engineering, thinkLets, and GSS design.

AgendaBuilder [29] assists novice facilitators in designing collaboration processes by helping them find and connect together the best thinkLets. This toolkit has the ability to filter the list of candidate thinkLets using the same methods as Kolfschoten, et al. [20]. It also has a rule base for each thinkLet based on contextual factors (e.g. group size, time, etc). The practitioner can use this rule base to determine if a thinkLet is a good match for the task at hand. AgendaBuilder provides no support for comparing thinkLets using these rules or for helping practitioners make trade-offs among these rules to find the best thinkLet for a task. Thus to find the best thinkLet, the practitioner must manually perform rough qualitative comparisons between thinkLets based on these rules.

There are also toolkits that use approaches other than collaboration engineering and thinkLets. None of these toolkits (e.g., [1,2,24,11]) have the capability to suggest all three of the collaboration elements presented earlier: a tool, its configuration, and a script. Toolkits such as those used by Aiken, et al. [1] and Antunes, et al. [2] have the ability to suggest the relevant tool but not the configuration or script. The toolkit by Antunes, et al. [2] helps the facilitator break the collaboration process into sub-tasks that fit into the collaboration patterns proposed by Kaner [17]. It then suggests a tool for each of those sub-tasks. Expert Session Planner (ESP) [1], which is designed to support facilitators during critical pre-session planning, has the ability to suggest a tool for a collaboration task based on answers to questions asked of the user.

In summary, there are two types of methods for guiding practitioners in the design of collaborative process. The first includes methods that do not use collaboration engineering techniques (e.g., [1,2,24,11])

and, thus, do not have the capability to leverage all three of the collaboration elements. The second type includes methods that make use of collaboration engineering techniques. These methods do not provide support for inexperienced practitioners to compare alternative thinkLets and make trade-offs.

We propose a design approach that helps practitioners compare and select different thinkLets without needing prior knowledge about collaboration engineering.

3. Design

A decision problem can be structured into three main components: a set of criteria or goals; a set of solutions or alternatives; and, consequences of the alternatives [13]. To design a successful collaboration process, the goals of the collaboration need to be understood and a decision needs to be made about which thinkLet (or combination of thinkLets) to use to achieve those goals. Each thinkLet has different consequences on the satisfaction of the group's goals and thus a decision about which thinkLet to use involves comparing alternatives and making trade-offs.

Since the problem of choosing the best thinkLet can be formulated as a MCDM problem, we use the nomenclature suggested by Stewart [28].

We define T_{set} to be the set of all thinkLets available from which a selection of a thinkLet $t \in T_{set}$ must be made (that is, T_{set} is the set of alternatives).

In addition to T_{set}, we define C_{set}, as the set of criteria by which elements of T_{set} are to be compared. These criteria can be identified by carefully reviewing the documentation of each thinkLet t in T_{set}. In addition, criteria that have been proposed in previous research for classifying thinkLets [18] can be used. Most of these criteria are related to group characteristics (e.g., the size of the group), meeting characteristics (e.g., the time that should be spent on a specific topic), or group goals and intention (e.g., producing creative ideas). For example if T_{set} = {OnePage, FreeBrainstorming} then C_{set} will be: {Number of people in the group, Time at hand}. This is because according to the documentation of these thinkLets [5], if less than 6 people are in the group or the team has less than 10 minutes to work on the activity, OnePage should be selected otherwise FreeBrainstorming is the better alternative. Every time a new thinkLet is added to the T_{set}, the criteria that are needed to compare that thinkLet with other thinkLets of T_{set} need to be added to C_{set}. For example, if we add FastFocus to T_{set} then C_{set} will become: {Number of people in the group, Time at hand, Pattern of collaboration}.

Finally, for each thinkLet t in T_{set}, we define a set of attributes $A^t = (a^t_{c1}, a^t_{c2}, \ldots, a^t_{cn})$ where n is the number of criteria in C_{set} and a^t_{ci} is the attribute representing the outcome of choosing thinkLet t on criterion c_i. For instance, for the thinkLet OnePage, and the criterion 'Pattern of collaboration',

$$a^{OnePage}_{Pattern\_of\_Collaboration} = \text{Diverge}$$

MCDM is a formal approach that can be used to solve problems that can be defined in terms of relatively precise sub-goals or criteria, which are generally conflicting [28]. MCDM techniques work best when the criteria are quantitative. In our case, quantitative measurements are not always appropriate for choosing among the alternatives because many of the factors that must be taken into consideration are qualitative (e.g., creative comments). MCDM methods that enable a decision maker to find the best alternative in the absence of quantitative data are very cumbersome even if there are only a few alternatives [12]. Since the selection space of thinkLets and the number of criteria are large and can grow without a bound [18], MCDM techniques cannot be used alone. Therefore, our model utilizes a decision tree as a classifier that first finds the best class of thinkLets that match the group's goals and then, if the class contains more than one thinkLet, uses MCDM techniques to find the best thinkLet among the alternatives within that class.

3.1. The decision tree

In our proposed decision tree, each leaf is a nonempty set of thinkLets. Internal nodes represent a point where criteria must be compared to make a choice between a number of attributes and each branch represents the outcome of choosing one of the possible attributes. As with any other decision tree, each path from the root to a leaf represents a classification rule. At each node the attributes being compared have either qualitative or quantitative values.

Most of the time, in order to achieve a team's goals, a collaboration process must be designed that consists of more than one thinkLet. Because the output resulting from thinkLet t limits the set of thinkLets that can be selected to follow t, and each decision starts at the root of the tree, there is a branch from the root for each possible previously selected thinkLet t' in T_{set} plus a branch that represents the case when no thinkLet has been selected previously. However, if there is no thinkLet in T_{set} that could follow t', there is no branch from the root for the case t' was previously selected. For instance if T_{set} ={FreeBrainstorming, FastFocus}, the root of the decision tree contains the following branches: {'No thinkLet was selected before', 'FreeBrainstorming was

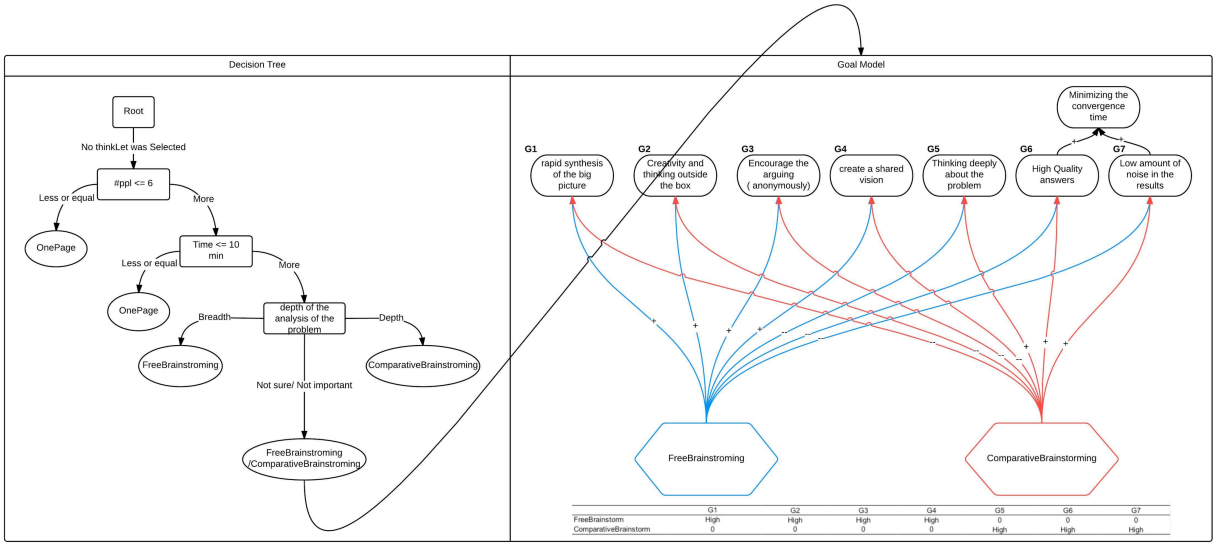

Figure1. An example of a decision tree and goal model for T_{set} = {OnePage, FreeBrainstorming, ComparativeBrainstorming}

selected before'}. Note that there is no 'FastFocus was selected before' branch, since it is not a good practice to send the output of the FastFocus activity directly to a FreeBrainstorming activity.

The decision tree model takes past research into consideration regarding appropriate sequences of thinkLets that have been identified [7,5]. For example, in order to reach consensus on the key issues of an unstructured list, it has been suggested that the sequence Fastfocus + StrawPoll + Crowbar is appropriate [5]. Thus, if 'reach consensus on the key issues of an unstructured list' is selected by the practitioner, this sequence of thinkLets should be suggested, rather than suggesting a single thinkLet (e.g., FastFocus alone). Figure 1 (left side) depicts an example of decision tree created for T_{set} = {OnePage, FreeBrainstorming, ComparativeBrainstorming}.

A 'not sure / not important' branch is included in the model to encourage the practitioner to postpone a decision if he/she is not sure which branch to take. For example he/she might not know in advance how many comments will be generated in a brainstorming activity, or he/she might not be able to decide if it is best to encourage people to think deeply about the solution (depth) or to push them for breadth and creative solutions. Note that in this situation both breadth and depth might be highly desirable which makes it harder for the practitioner to decide. In these situations MCDM techniques can be used to help practitioners make trade-offs.

3.2. MCDM and goal programming

Each leaf of the decision tree either contains a single thinkLet or a set of possible thinkLets. In some cases, the criteria evaluated at each node in the decision tree will result in a path to a leaf with a single thinkLet that can be selected. In this case, the best thinkLet is found using only the classification rules; however, if the leaf contains a set of thinkLets, then a choice must be made among possibly conflicting goals (e.g., producing high quality ideas vs. producing creative ideas). In these situations, MCDM techniques can be used to guide the practitioner select a single, best thinkLet from the set.

Suppose there is a leaf node called '*leaf*', that contains a set T_{leaf}, $T_{leaf} \subset T_{set}$ of more than one thinkLet. We define C_{leaf}, $C_{leaf} \subset C_{set}$, as the set of criteria or goals that can be used to compare the alternative thinkLets in T_{leaf}. Also, for each thinkLet t in T_{leaf}, we define a set of attributes $A_{leaf}^t = (a^t_{c1}, a^t_{c2}, \ldots, a^t_{ck})$ where $k < n$ is the number of criteria in C_{leaf} and a^t_{ci} is the attribute representing the outcome of choosing thinkLet t on criterion c_i. Note that A_{leaf}^t is a subset of the A^t because $C_{leaf} \subset C_{set}$.

For each T_{leaf} where $|T_{leaf}| > 1$, in the decision tree, a goal model is created that captures the positive and negative contribution of each t in T_{leaf} on each goal in C_{leaf}. Figure 1 (right side) shows the goal model created for T_{leaf} = {FreeBrainstorming, ComparativeBrainstorming} where T_{set} = {OnePage, FreeBrainstorming, ComparativeBrainstorming}.

Elahi et al. use goal modeling notation to represent criteria and alternatives in the form of a decision problem [12]. Their notation consists of two main modeling elements: solutions (or alternatives) and goals (or criteria). We propose to use the same approach and notation as [12] to create a goal model for each T_{leaf}.

The challenge that we are facing is that in order to be able to choose the best thinkLet in T_{leaf}, we first

need to determine the value of the elements of A_{leaf}^{t} for each thinkLet *t* in T_{leaf}. But since the criteria or goals in C_{leaf} are often intangible, assigning a value to each attribute is a non-trivial task. The *comparing the alternative* method can be used to give a value between 0-5 to each attribute in A_{leaf}^{t}. That is, for each criteria c_i in C_{leaf}, if thinkLet *t* maximally satisfies c_i, the value of a^{t}_{ci} is 5. Correspondingly, for each thinkLet *t'* in T_{leaf}, the value of $a^{t'}_{ci}$ is assigned a value relative to the maximally-satisfying attribute value a^{t}_{ci}. The attribute values can be determined from the thinkLet's documentation. It is worth mentioning that it is possible for more than one thinkLet to have the same value for an attribute that satisfies a specific goal. For example, if thinkLets *t* and *t'* both maximally satisfy the goal c_i, then $a^{t}_{ci} = a^{t'}_{ci} = 5$.

Comparing alternatives has been recognized as a powerful approach when a decision maker needs to work with conflicting and intangible criteria which, by definition, have no scales of measurement [27]. This approach is the basis of several MCDM methods such as ratio pair-wise comparison [26] and Simple Multi-Attribute Rating Technique (SMART) [30]. Our method for determining the value of the attributes is based on the direct rating technique, which can be classified as a SMART method.

We propose using a scale for comparison that has been suggested in [12]. This scale consists of 6 levels for distinguishing the decision elements: Zero (0); Low (1); Medium Low (2); Medium (3); Medium High (4); and, High (5). In this scale, 0 is the minimum satisfaction level for a goal and 5 is the maximum satisfaction level for a goal. The table in Figure 1 shows the values that were assessed for the attributes of thinkLets FreeBrainstorming and ComparativeBrainstorming (as determined from the thinkLet documentation).

The goals associated with each T_{leaf} are assigned weights that must be determined according to the given collaboration needs and so the practitioner must assign weights to each of the goals. To do this, the practitioner needs to rate the criteria in C_{leaf} using a 5-point Likert-type scale, where 1 means satisfying the criterion (or goal) has the lowest priority and 5 means that satisfying the criterion has the highest priority. Rating is an example of direct assessment weighting and can be used to determine the priority of each criteria based on the decision maker's preferences [16].

After each criterion c_i in C_{leaf} has been assigned a weight w_i, the preemptive goal-programming rule combined with an additive value function can be used to find the best alternative (Figure 2 shows a step-by-step example).

To begin, a list of candidate solutions is defined called T_{candid}. Initially, $T_{candid} = T_{leaf}$. In the first step, all criteria with the same weight are grouped together. Next, the groups are ordered by their weights (or their priority) such that the group with the highest priority is GR_1 and the group with the lowest priority is GR_z where z is the number of groups.

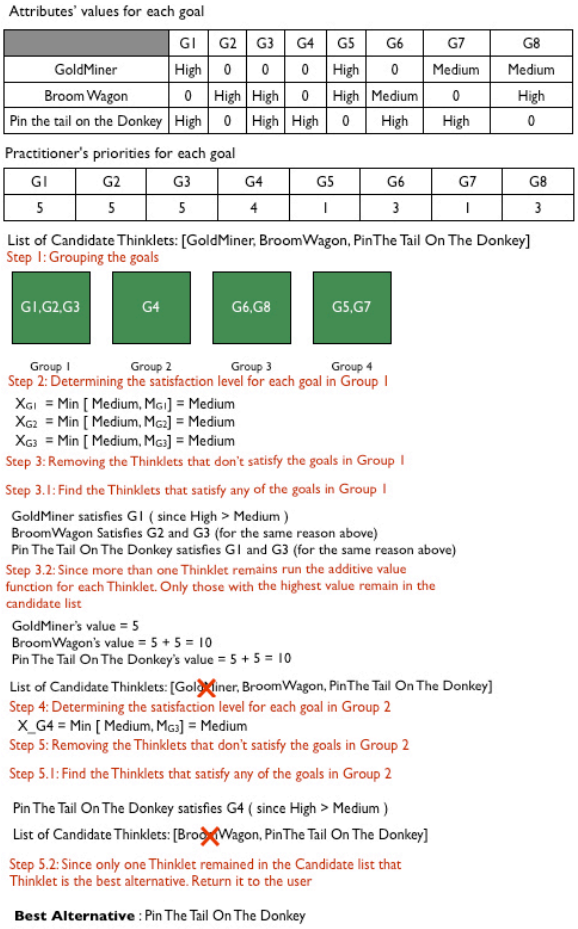

Figure 2. Choosing the best thinkLet using preemptive goal programming and an additive value function.

Then, for each of the criteria in GR_1, a level of satisfaction is determined in the following way. In each iteration the satisfaction value of criterion c_i is dynamically defined as the Minimum of [Medium, M_{ci}], where M_{ci} is the maximum value of the a^{t}_{ci}. If a thinkLet *t* in T_{leaf} satisfies none of the criteria in the first group, then *t* is removed from T_{candid}.

After removing the thinkLets that do not satisfy any goal of the first group, if more than one thinkLet remains in T_{candid}, the additive value function is used to remove solutions from T_{candid} that do not maximize the value of this function. Note that only the attributes of the thinkLets associated with the goals in the first group (the group with the highest priority) are used in this function. In other words, if $C_{GR1} \subset C_{leaf}$, (where

$|C_{GR1}| = p$), is the group with criteria of highest priority, the additive value function is defined as:

$$Value_t = \sum_{i=1}^{p} a_{c_i}^{t}$$

where t is in T_{candid}, and a_{ci}^{t} is the attribute in A_{leaf}^{t} which is associated with the goal c_i in C_{G1} . Note that there is no weight in this formula since in this group all the criteria have the same weight.

This process is repeated for each priority group in turn, until the list of the size of T_{candid} is reduced to one, or there are no more criteria left. It is very unlikely that the latter case will happen because the thinkLets typically have very different attribute values. But in the case that there are no criteria left and more than one thinkLet remains in T_{candid}, any one of the thinkLets that remain in T_{candid} can be chosen at random as a solution.

4. Evaluation

In order to evaluate our approach, we created a prototype, CP_Dez, that enables a user to navigate through the full set (T_{set}) of 39 thinkLets in [5], making trade-offs among different goals using the multi-criteria decision making (MCDM) techniques described in Section 3.2. Navigation of the thinkLets is supported by a user interface (UI) that provides access to the underlying decision-tree model described in Section 3.1 (see Figure 3).

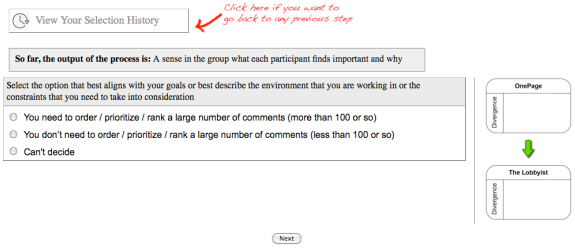

Figure 3. CP-Dez's UI

Each selection determines the edge that will be traversed in the underlying decision tree. At each subsequent node of the decision tree, prompts are presented to either gather choices among alternatives (e.g., Figure 3) or, if a leaf with more than one thinkLet is reached, to gather rating information about the importance of specific goals or constraints (e.g., Figure 4). In the latter case, ratings are needed to provide input to the MCDM process in order to choose a thinkLet from a class of thinkLets.

The result of this procedure will be a selected thinkLet or a sequence of thinkLets. Once a thinkLet is selected, it is displayed on the right hand side of the interface (along with any previously selected thinkLets in the sequence) and the practitioner is prompted to indicate if they are satisfied with this final result or if they would like to continue with the task of designing the collaboration process. This process is repeated, each time starting at the root node, until satisfaction with the final sequence of thinkLets is indicated.

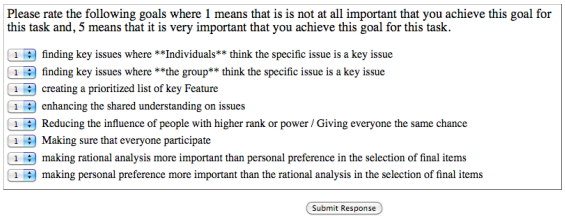

Figure 4. The participants are asked to rate criteria when the leaf contains a set of more than one thinkLet

Our evaluation was carried out through an experiment in which participants were asked to design a collaboration process for a scenario whereby a consortium of universities wish to revise its joint strategy. We chose this scenario from a paper in the collaboration engineering literature [4] because the design presented in the paper can be used as the "answer key" or "ground truth" result to which we can compare the quality of the designs produced by our participants. Furthermore, this multi-organization strategy development scenario can be understood by participants with different domain expertise. It also requires tasks from all of the different collaboration patterns, which can demonstrate that the prototype can be used to select thinkLets from different patterns of collaboration categories. In [4], the scenario was broken into 12 tasks. For the purposes of our experiment, we selected seven of the 12 tasks and omitted five that used the same thinkLets as the other seven tasks.

We asked eighteen participants to design a collaboration process for the scenario above. The participants were recruited using convenience sampling and had a variety of domain expertise including: science and engineering, business, education, management, and human resources. Half of the participants used our prototype (CP-Dez) to design the collaboration process and the other half used the thinkLet book [5]. We used a between-subjects study design with one independent variable: the tool that subjects used to design the collaboration process. This independent variable had 2 possible levels: book vs. CP-Dez. The dependent variables in this study were time and quality of the design. The efficiency of the design process was also measured using the quality

over time. These variables were used to test the following hypotheses: people who use the tool will be able to design H1) a higher quality collaboration process for that scenario in H2) a shorter amount of time and H3) more efficiently than those who use the book.

We identified work experience as a confounding variable and handled the effect of it by using a block design; that is, two experience blocks were defined. The first block contained current graduate students or recent graduates (all with less than 3 years of work experience) and the second block contained experienced participants who had more than 5 years of work experience. The treatments (book or CP-Dez) were randomly assigned to the participants in each block. The participants were recruited based on their work experience such that an equal number of participants (nine) was in each block. Overall, five experienced and four inexperienced participants used the book and the rest used CP-Dez.

We created a web application that guided each participant through the study. The application starts by asking participants to participate in a short (15 minutes) mandatory interactive training session on the basics of collaboration engineering, such as task breakdowns and patterns of collaboration. After the training session, seven tasks from the multi-organization strategy development scenario were given to the participants who were asked to design a collaboration process by selecting a thinkLet or sequence of thinkLets for each of these tasks (See Appendix A). The participants who were assigned to work with the thinkLet book had to look for the right thinkLets in the book and type the thinkLets' names into the text boxes that were provided in the web application. The CP-Dez interface was embedded in the application for those who were assigned to work with it and the suggested thinkLets that were accepted by the participants were automatically recorded as the answer.

In order to measure the quality of the designs, we compared the input, output, and criteria of each individual thinkLet selected by participants with those in the ground truth design. In addition, for each combination of two consecutive thinkLets, we determined if the combination was Tricky, Excellent, or Impossible [19]. We assigned quality points to the thinkLet(s) chosen for each task. Details of the point breakdown for a given task can be found in Appendix A. The selected thinkLets were evaluated and points assigned based on information about the ground truth in [4] and from descriptions of thinkLets in [5].

The time taken to complete the designs by each participant was also measured and the efficiency of the design process was calculated as quality over time.

5. Results

Figure 5, 6, and 7, respectively show the average quality of the designs, average time taken to complete the design of all tasks, and the efficiency of the design procedure for participants who used CP-Dez and the book. For clarity, in these figures, we have separated the scores of experienced and inexperienced participants for each treatment.

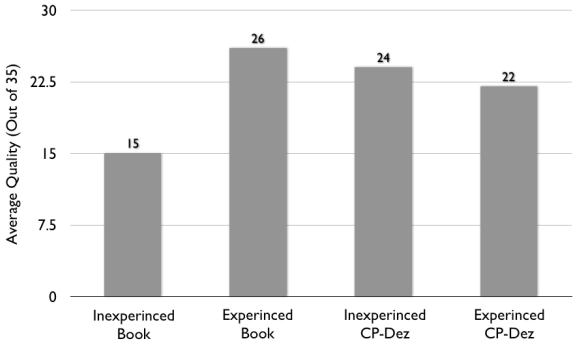

Figure 5. Comparison of the avg quality

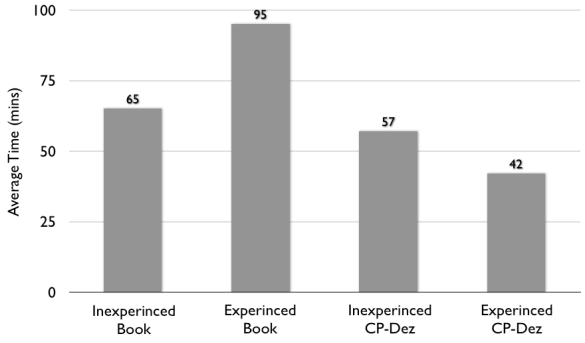

Figure 6. Comparison of the avg time spent

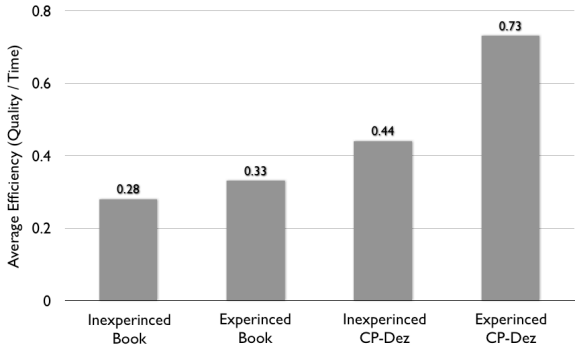

Figure 7. Comparison of the avg efficiency

Because of our small sample size, we used the nonparametric Mann-Whitney test to determine the significance of the results and tested our two-tailed hypotheses. We found that there is no significant difference between the qualities of the collaboration

processes designed using CP-Dez or the thinkLet book. We also found that the average time taken to complete the design was not significant for participants who used CP-Dez or the thinkLet book. However a Mann-Whitney test indicated that the efficiencies of the design process was significantly greater for participants who used CP-Dez (Mdn = 0.48) than for participants who used the thinkLet book (Mdn = 0.29), $U = 17$, $p = 0.042$. This indicates that in a unit of time, participants created higher quality designs when using CP-Dez.

The resulting thinkLets chosen for the tasks highlight some of CP-Dez's strengths and weaknesses. For example, the solution in [4] proposed the Concentration thinkLet for merging and generalizing the statements of a list. In the book, this thinkLet is listed under the Organize pattern; however, it can also be used to produce a Converge-like pattern, for example, to remove the redundancy in a list. In CP-Dez, this thinkLet can be found in paths that include an Organize or Converge branch. According to our results, most of the participants considered this task to be a Converge task (16/18). In fact, 8/9 of the participants who used CP-Dez decided that the Converge attribute was the best value that describes this task's pattern of collaboration and 6/8 of them found the Concentration thinkLet as the match for the task. However, only 1 out of 9 participants who used the book selected it as a fit for accomplishing the task. This is one of the strengths of CP-Dez. When the borders between the categories are fuzzy for some alternatives, the alternatives can be shown under all of those categories and the practitioner can make the decision based on other important criteria.

The results of another task in our study can be used to illustrate another strength of CP-Dez. In this task the goal of the collaboration was to prioritize the challenges of having an information system that is shared between multiple universities. This complicated task requires a sequence of thinkLets. Inexperienced participants using the book performed very poorly on this task (the quality score for all of them was negative for this task). However, the inexperienced participants who used CP-Dez performed very well on this task (average quality score of 3.4/5 for this task). For this particular task, inexperienced participants who used CP-Dez performed better than the experienced participants who used the book (whose average score was 2.4/5 for this task).

One of the weaknesses of CP-Dez is that when a participant cannot find an option that matches with what he/she wants to do, it is very difficult to choose the next step and find the best thinkLet. Even though the History button was used a lot during the study (all of the participants who used CP-Dez went back and changed their answers at least twice), there were still cases in which participants had trouble choosing the correct option. In the future we plan to give practitioners access to the pool of candidate thinkLets. This will enable more experienced people to manually select a thinkLet from among the remaining candidates at any step. This feature can help reduce the time needed for the selection process if the practitioner is confident that he/she can manually select a thinkLet or if he/she does not agree with a suggested thinkLet and wants to change that suggestion.

6. Conclusion

In this paper, we presented an approach which can be used to help practitioners in designing collaboration processes using thinkLets. Our study suggests that this approach significantly improves the efficiency of the design procedure. In our approach, the model can be easily expanded and maintained, by adding the new comparison criteria to the existing decision tree, updating the leaves, and adding/updating necessary goal models as described in section 3. This model highlights small differences between thinkLets which might not be obvious to practitioners or inexperienced facilitators. In addition it can be useful even to more experienced facilitators who need to make trade-offs between different thinkLets in more complicated situations where a team needs to reach several qualitative goals with different priorities.

Our preliminary study was a double-blind experiment. That is, the scenario was chosen after the creation of the CP-Dez's model and no fine-tuning was performed on the model after the scenario was chosen. This could help avoid the experimenters' tendency to fine-tune the model to work better with the chosen scenario. Although the scenario is carefully chosen so that it contains all different patterns of collaboration, it cannot fully support the claims of generalizability (this is a threat to external validity). Additional studies should be carried out using different scenarios in which experts evaluate the designs created by CP-Dez instead of comparing designs to a ground truth. It would also be interesting to evaluate not only the collaboration process design produced using CP-Dez but the results of having novice facilitators carry out the suggested collaboration process to see if the design created by CP-Dez lead to an effective decision-making process.

While creating our prototype, although the process of reviewing the thinkLet documentation to identify the attributes of the thinkLets and the criteria for comparing thinkLets was performed systematically, one of the major limitations of this work is the subjectivity involved in doing so. In the future, a community of experts should verify the attributes and

criteria. We are planning to create a platform that enables the community of collaboration engineers to create, maintain, and update a unified model. This model can then be used as is or can be customized for a specific organization's needs. In this platform, branches of the tree, and goal models can be modified based on the group's consensus.

7. Acknowledgements

This research was supported by an NSERC Collaborative Research and Development Grant with SAP.

8. References

[1] Aiken, M. W., Motiwalla, L. F., Sheng, O. L., & Nunamaker Jr, J. F. (1990, January). "ESP: An expert system for pre-session group decision support systems planning." In System Sciences, 1990., Proceedings of the Twenty-Third Annual Hawaii International Conference on, vol. 3, pp. 279-286. IEEE, 1990.

[2] Antunes, P., Ho, T., & Carriço, L. "A GDSS agenda builder for inexperienced facilitators." Proceedings of the 10th EuroGDSS workshop. Copenhagen, Denmark: Delft University of Technology, 1999.

[3] Bajwa, D. S., Lewis, L. F., Pervan, G., & Lai, V. S. "The adoption and use of collaboration information technologies: International comparisons." Journal of Information Technology 20, no. 2 (2005): 130-140.

[4] Bragge, J., Merisalo-Rantanen, H., Nurmi, A., & Tanner, L. "A repeatable e-collaboration process based on thinklets for multi-organization strategy development." Group Decision and Negotiation 16, no. 4 (2007): 363-379.

[5] Briggs, R. O., & De Vreede, G. J. thinkLets: building blocks for concerted collaboration. University of Nebraska, Center for Collaboration Science, 2009.

[6] Briggs, R. O., De Vreede, G. J., & Nunamaker Jr, J. F. "Collaboration engineering with thinkLets to pursue sustained success with group support systems." J. of Management Information Systems 19.4 (2003): 31-64.

[7] Briggs, R. O., De Vreede, G., Nunamaker Jr, J. F., & Tobey, D. "ThinkLets: achieving predictable, repeatable patterns of group interaction with group support systems (GSS)." In System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference on, pp. 9-pp. IEEE, 2001.

[8] Clawson, V. K., Bostrom, R. P., & Anson, R. "The role of the facilitator in computer-supported meetings." Small Group Research 24.4 (1993): 547-565.

[9] De Vreede, G. J., & Briggs, R. O. "Collaboration engineering: designing repeatable processes for high-value collaborative tasks." System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on. IEEE, 2005.

[10] De Vreede, G. J. "A field study into the organizational application of group support systems." Journal of Information Technology Case and Application Research 2, no. 4 (2000): 27-47.

[11] Dickson, G. W., Partridge, J. E. L., & Robinson, L. H. "Exploring modes of facilitative support for GDSS technology." MIS Quarterly (1993): 173-194.

[12] Elahi, G., & Yu, E."Comparing alternatives for analyzing requirements trade-offs–In the absence of numerical data." Information and Software Technology 54.6 (2012): 517-530.

[13] Figueira, J., Greco, S., & Ehrgott, M. *Multiple criteria decision analysis: state of the art surveys.* Vol. 78. Springer, 2005.

[14] Fjermestad, J., & Hiltz, S. R. "Group support systems: A descriptive evaluation of case and field studies." Journal of Management Information Systems 17.3 (2000): 115-160.

[15] Grohowski, R., McGoff, C., Vogel, D., Martz, B., & Nunamaker, J. (1990). "Implementing electronic meeting systems at IBM: lessons learned and success factors." Mis Quarterly (1990): 369-383.

[16] Hobbs, B. F., & Meier, P. M. "Multicriteria methods for resource planning: an experimental comparison." Power Systems, IEEE Transactions on 9.4 (1994): 1811-1817.

[17] Kaner, S. *Facilitator's guide to participatory decision-making.* John Wiley & Sons, 2007.

[18] Kolfschoten, G. L., Briggs, R. O., Appelman, J. H., & de Vreede, G. J. "ThinkLets as building blocks for collaboration processes: a further conceptualization." In Groupware: Design, Implementation, and Use, pp. 137-152. Springer Berlin Heidelberg, 2004.

[19] Kolfschoten, G. L., & De Vreede, G. J.. "The collaboration engineering approach for designing collaboration processes." Groupware: design, implementation, and use. Springer Berlin Heidelberg, 2007. 95-110.

[20] Kolfschoten, G. L., & Veen, W."Tool support for GSS session design." System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on. IEEE, 2005.

[21] Moradian, A., Nasir, M., Lyons, K., Leung, R., & Sim, S. E. (2014, April). "Gamification of collaborative idea generation and convergence." In CHI'14 Extended Abstracts on Human Factors in Computing Systems, pp. 1459-1464. ACM, 2014.

[22] Munkvold, B. E., & Zigurs, I. "Integration of e-collaboration technologies: Research opportunities and challenges." International Journal of e-Collaboration (IJeC) 1.2 (2005): 1-24.

[23] Nunamaker, J. F., Briggs, R. O., Mittleman, D. D., Vogel, D. R., & Balthazard, P. A. (1997). "Lessons from a dozen years of group support systems research: A discussion of lab and field findings." Journal of management information systems (1996): 163-207.

[24] Nunamaker, J. F., Dennis, A. R., Valacich, J. S., Vogel, D., & George, J. F. "Electronic meeting systems." Communications of the ACM 34, no. 7 (1991): 40-61.

[25] Post, B. Q. "A business case framework for group support technology." Journal of Management Information Systems 9.3 (1992): 7-26.

[26] Saaty, T. L. "A scaling method for priorities in hierarchical structures." Journal of mathematical psychology 15.3 (1977): 234-281.

[27] Saaty, T. L. "The analytic hierarchy and analytic network processes for the measurement of intangible criteria and for decision-making." Multiple criteria decision analysis: state of the art surveys. Springer New York, 2005. 345-405.

[28] Stewart, T. J. "A critical survey on the status of multiple criteria decision making theory and practice." Omega 20.5 (1992): 569-586.

[29] Vivacqua, A. S., Ferreira, M. S., & de Souza, J. M. "AgendaBuilder: A system to support meeting design." In Computer Supported Cooperative Work in Design (CSCWD), 2010 14th International Conference on, pp. 596-601. IEEE, 2010.

[30] Von Winterfeldt, D., & Edwards, W. *Decision analysis and behavioral research*. Vol. 604. Cambridge: Cambridge University Press, 1986.

Appendix A:

In this Appendix, the scenario and one of the tasks broken out of the scenario are presented. The correct answer (ground truth or answer key) from [4] is given and the metrics (point system) that was used to measure the closeness of participant answers to the correct answer are explained. The maximum score is 5.

Scenario: The team tries to create an agenda for a six hour process of revising the strategy of the consortium of 13 Finnish universities in charge of a joint student information system (IS). This information system serves three user groups: student administration, students, and teachers. There are 18 team members in this session (ten represent student administration, seven represent IT services and one represents the consortium). Each team member has their own computer. The team wants to mainly focus on creativity tasks without any pressure to make final decisions. Also, democratic involvement of the team members and equality is very important (anonymity is preferred).

Example Task: In the first step of the collaboration process, the team members need to identify the challenges related to having a joint information system. The time allocated for this task is 35 minutes and it is expected that a large amount of ideas will be generated.

Correct Answer (from [4]): FreeBrainstorming (Diverge)

Table 1 explains how we measure the quality of the answers given by study participants:

Point	Criteria	Additional Comments for this specific example
1	The collaboration pattern is chosen correctly	The process should contain a Diverge thinkLet
1	The same thinkLet as the ground truth thinkLet or a thinkLet that has the same criteria is chosen	FreeBrainstorming thinkLet should be chosen
1	The Diverge thinkLet is followed by a convergence or elaboration thinkLet	
-1	The combination of thinkLets proposed is "impossible" (see section 1 for the definition of an impossible combination)	
-0.5	The combination of thinkLets proposed is "tricky" (see section 1 for the definition of a tricky combination)	
1	The thinkLets are matched with task's specifications and inputs	In this example: * The input should be a single question * more than 6 people are in the team * The time is more than 10 minutes * Creativity is important * Anonymity is important
1	The output of the sequence of thinkLets can be used to accomplish the task	The sequence should at some point produce a large set of brainstorming comments
-1	There is a redundant thinkLet in the sequence	
-1	The sequence of thinkLets or the thinkLet should not be used in these situations	Use the documentation of the thinkLets to judge

Table 1. Measuring scheme for quality of answers